chapter 2

General Security Concepts

From one thing, know ten thousand things.

— Miyamoto Musashi

In this chapter, you will learn how to

- Define basic terms associated with computer and information security
- Identify the basic approaches to computer and information security
- Identify the basic principles of computer and information security
- Recognize some of the basic models used to implement security in operating systems

In Chapter 1, you learned about some of the various threats that we, as security professionals, face on a daily basis. In this chapter, you start exploring the field of computer security. Computer security has a series of fundamental concepts that support the discipline. We begin with an examination of security models and concepts and then proceed to see how they are operationally employed.

Basic Security Terminology

The term **hacking** has been used frequently in the media. A hacker was once considered an individual who understood the technical aspects of computer operating systems and networks. Hackers were individuals you turned to when you had a problem and needed extreme technical expertise. Today, primarily as a result of the media, the term is used more often to refer to individuals who attempt to gain unauthorized access to computer systems or networks. While some would prefer to use the terms *cracker* and *cracking* when referring to this nefarious type of activity, the terminology generally accepted by the public is that of hacker and hacking. A related term that may sometimes be seen is **phreaking**, which refers to the "hacking" of the systems and computers used by a telephone company to operate its telephone network.

The field of computer security constantly evolves, frequently introducing new terms that are often coined by the media. Make sure to learn the meaning of terms such as *hacking, phreaking, vishing, phishing, pharming,* and *spear phishing*. Some of these terms have been around for many years (such as hacking), whereas others have appeared only in the last few years (such as spear phishing).

Security Basics

Computer security itself is a term that has many meanings and related terms. Computer security entails the methods used to ensure that a system is secure. Subjects such as authentication and access controls must be addressed in broad terms of computer security. Seldom in today's world are computers not connected to other computers in networks. This then introduces the term *network security* to refer to the protection of the multiple computers and other devices that are connected together. Related to these two terms are two others—*information security* and *information assurance*—that place the focus of the security process not on the hardware and software being used but on the data that is processed by them. Assurance also introduces another concept: that of the availability of the systems and information when we want them. The common press and many professionals have settled on *cybersecurity* as the term to describe the field. Still another term that may be heard in the security world is COMSEC, which stands for *communications security* and deals with the security of telecommunication systems.

Cybersecurity has become regular headline news these days, with reports of break-ins, data breaches, fraud, and a host of other calamities. The general public has become increasingly aware of its dependence on computers and networks and consequently has also become interested in the security of these same computers and networks. As a result of this increased attention by the public, several new terms have become commonplace in conversations and print. Terms such as *hacking, viruses, TCP/IP, encryption,* and *firewalls* are now frequently encountered in mainstream news media and have found their way into casual conversations. What was once the purview of scientists and engineers is now part of our everyday life.

With our increased daily dependence on computers and networks to conduct everything from making purchases at our local grocery store to banking, trading stocks, receiving medical treatment, and driving our children to school, ensuring that computers and networks are secure has become of paramount importance. Computers and the information they manipulate have become a part of virtually every aspect of our lives.

Tech Tip

CIA of Security

While there is no universal agreement on authentication, auditability, and nonrepudiation as additions to the original CIA of security, there is little debate over whether confidentiality, integrity, and availability are basic security principles. Understand these principles, because one or more of them are the reason most security hardware, software, policies, and procedures exist.

The "CIA" of Security

Almost from its inception, the goal of computer security has been threefold: confidentiality, integrity, and availability—the "CIA" of security. The purpose of **confidentiality** is to ensure that only those individuals who have the authority to view a piece of information may do so. No unauthorized individual should ever be able to view data they are not entitled to access. **Integrity** is a related concept but deals with the generation and modification of data. Only authorized individuals should ever be able to create or change (or delete) information. The goal of **availability** is to ensure that the data, or the system itself, is available for use when the authorized user wants it.

As a result of the increased use of networks for commerce, two additional security goals have been added to the original three in the CIA of security. **Authentication** attempts to ensure that an individual is who they claim to be. The need for this in an online transaction is obvious. Related to this is **nonrepudiation**, which deals with the ability to verify that a message has been sent and received and that the sender can be identified and verified. The requirement for this capability in online transactions should also be readily apparent. Recent emphasis on systems assurance has raised the potential inclusion of the term **auditability**, which refers to whether a control can be verified to be functioning properly. In security, it is imperative that we can track actions to ensure what has or has not been done.

The Fortress Model

The original model for computer security was the **fortress model**—keep the bad out, allow in the good. This was a natural model: build a series of defenses and your system can be secure. This also led to the question "Are we secure?" For this question, a binary yes-or-no answer was expected. However, time has shown that this model is not realistic. A new version of the fortress model, called endpoint security, involves securing of all endpoints in a network are secured from all threats. Not a day goes by in the security industry where some pundit declares that endpoint security is dead. When viewed as the only defense, endpoint security leaves a lot to be desired, but as part of an overall program, endpoint security still plays a valuable role. The fortress model has been shown to not provide sufficient defenses, yet, like endpoint security shows, it is still a valuable component in a modern security program.

The Operational Model of Computer Security

For many years, the focus of security was on prevention. If we could prevent everyone who did not have authorization from gaining access to our computer systems and networks, then we assumed that we had achieved security. Protection was thus equated with prevention. While the basic premise of this is true, it fails to acknowledge the realities of the networked environment our systems are part of. No matter how well we seem to do in prevention technology, somebody always seems to find a way around our safeguards. When this happens, our system is left unprotected. Thus, we need multiple prevention techniques and also technologies to alert us when prevention has failed and to provide ways to address the problem. This results in a modification to our original security equation with the addition of two new

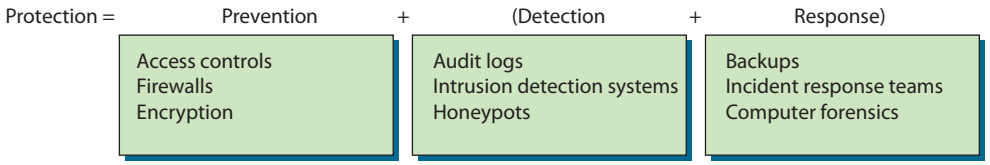


Figure 2.1 Sample technologies in the operational model of computer security

elements: detection and response. Our security equation thus becomes the following:

Protection = Prevention + (Detection + Response)

This is known as the **operational model of computer security**. Every security technique and technology falls into at least one of the three elements of the equation. Examples of the types of technologies and techniques that represent each are depicted in Figure 2.1.

Time-Based Security

In 1998, Winn Schwartau published a paper that was clearly ahead of its time. The paper was on the topic of time-based security. Time-based security was not a new concept; in fact, it has been used in physical security for years. Safes are rated in terms of how long they will resist penetration. Bringing the concept of time to the operational security model puts it in line with modern security defense practices. Time-based security allows us to understand the relationship between prevention, detection, and response. The Schwartau paper uses the term *protection* for prevention and the term *reaction* for response. The following is from the Schwartau paper:

> Information security is now a simple affair. You no longer have to build up huge levels of protection. You need to concentrate on the detection and reaction. . . which ultimately determines the amount of effective security you have.

Simply put, the amount of time offered by a protection device, P_t, should be greater than the time it takes to detect the attack, D_t, plus the reaction time of the organization, R_t:

$$P_t > D_t + R_t$$

The remainder of the paper is devoted to the discussion of how to use time-based security to make economic security decisions. One of these key decisions, borne from the arguments in the paper, is whether particularly sensitive information belongs on the network to begin with. "Sometimes, some information has no business being on a network. The cost of protection, versus the downside risk, is just not worth it." This paper describes how we do security today—threat intelligence, kill chains, incident response, and a host of options beyond simple fortress foundations such as firewalls.

Cybersecurity Framework Model

In 2013, President Obama signed an executive order directing the U.S. National Institute of Science and Technology (NIST) to work with industry and develop a cybersecurity framework. This was in response to several significant cybersecurity events where the victim companies appeared to be

Tech Tip

NIST Cybersecurity Framework

The NIST Cybersecurity Framework is a risk-based approach for the implementation of cybersecurity activities in an enterprise. The framework provides a common taxonomy of standards, guidelines, and practices that can be employed to strengthen cybersecurity efforts. To learn more about the NIST Cybersecurity Framework, see https://www.nist.gov/cyberframework.

unprepared. The resultant framework, titled *Framework for Improving Critical Infrastructure Cybersecurity*, was created as a voluntary system, based on existing standards, guidelines, and practices, to facilitate adoption and acceptance across a wide array of industries.

The NIST Cybersecurity Framework provides a common taxonomy and mechanism to assist in aligning management practices with existing standards, guidelines, and practices. Its purpose is to complement and enhance risk management efforts through the following actions:

1. Determining the current cybersecurity posture
2. Documenting the desired target state with respect to cybersecurity
3. Determining and prioritizing improvement and corrective actions
4. Measuring and monitoring progress toward goals
5. Creating a communication mechanism for coordination among stakeholders

The framework is composed of five core functions, as illustrated in Figure 2.2. Two of these core functions, *Identify* and *Protect*, describe actions taken before an incident. *Detect* is the core function associated with intrusion detection or the beginning of an incident response. The last two, *Respond* and *Recover*, detail actions that take place during the post-incident response. Examples of the items under each function are illustrated in the figure. In addition to the five functions, the framework has levels of implementations referred to as *tiers*. These tiers represent the organization's ability, from Partial (Tier 1) to Adaptive (Tier 4).

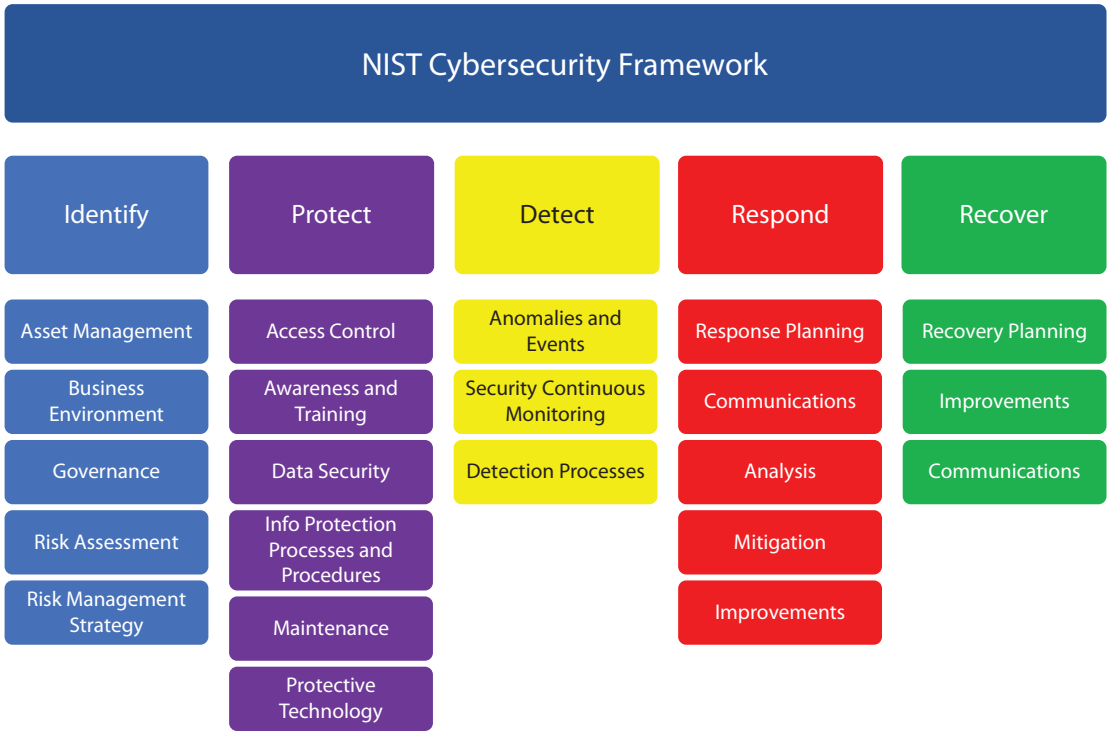


Figure 2.2 Cybersecurity Framework core functions

Active Defense Model

A new form of security mode is one based on active defense. While most of the models previously discussed focus on "static" defenses that are enacted to act as a barrier to intruders, active defense goes to the next level, the actual hunting of intruders inside the enterprise. This model capitalizes on elements of both the operational model and time-based security models. It makes the assumption that the static defenses, while necessary, are not sufficient. And as a result, some intruders will succeed in getting into the enterprise. The active defense model is one built around the actions necessary to actively seek out attackers that make it past the defenses. Similar to incident response procedures, but not dependent upon a specific incident, active hunters use their knowledge of baseline conditions for the systems and search for things that are abnormal. When abnormalities are found, they are chased to ground, and if an intruder is found, actions are taken to close the holes used to gain entry.

Security Tenets

In addition to the CIA elements, there are tenets that form a basis for system security. The three operational tenets found in secure deployments are session management, exception management, and configuration management.

Tech Tip

Session Management Cheat Sheet

Session management is a common task for web applications, and the Open Web Application Security Project (OWASP) has a cheat sheet to assist in the correct implementation of session management. See https://www.owasp.org/index.php/Session_Management_Cheat_Sheet.

Session Management

Session management is the set of activities employed to establish a communication channel between two parties, identifying each in a manner that allows future activity without renewed authentication. Session management allows an application to authenticate once and have subsequent activities ascribed to the authenticated user. Sessions are frequently used in web applications to preserve state and user information between normally stateless clicks.

Sessions are typically identified by an ID that is known to both sides of the conversation. This ID can be used as a token for future identification. If confidentiality is required, then the channel should be secured by an appropriate level of cryptographic protection.

Session management includes all the activities necessary to manage the session—from establishment, during use, and at completion of the conversation. Because the session represents the continuity of a security condition established during authentication, the level of protection that should be afforded to the session ID should be commensurate with the level of security initially established.

Exception Management

Exceptions involve the invocation of conditions that fall outside the normal sequence of operation. Whether by error or malicious action, exceptions are changes to normal processing and need to be managed. The special processing required by conditions that fall outside normal parameters can result in errors either locally or in follow-on processes in a system. The handling of

exceptions, referred to as *exception handling*, is an important consideration during software development.

Exception management is more than just exception handling in software development. When the operation of a system encounters an exception, whether it is invoked by a person, process, technology, or combination thereof, the system must effectively handle the condition. This can mean many different things—sometimes even operating outside normal policy limits. Exception management can also be nontechnical in nature: systems or environments that cannot follow organizational security policy, for example, must be documented, exceptions must be approved, and mitigations must be put in place to lower the risk associated with exceptions to policy. The bottom line is simple: either the system must handle the condition and recover or it must fail and be recovered by separate action. Designing exception handling into a system makes it more resilient, because exceptions will happen, and how they are handled is the only unknown outcome.

Configuration Management

Configuration management is key to the proper operation of IT systems. IT systems are first and foremost systems—groups of elements that work together to achieve a desired resultant process. The proper configuration and provisioning of all the components in a system is essential to the proper operation of the system. The design and operation of the elements to ensure the proper functional environment of a system is referred to as *configuration management*. Configuration management is a key operation principle and is thoroughly covered in Chapter 21.

Tech Tip

Got Network?

A classic black T-shirt in the security industry says "got root?" It's a takeoff on the successful ad campaign "got milk?" and indicates the power of root privilege. Similar to "got root?" is "got network?"; if you truly "own" the network, then you have significant control over what passes across it, which can result in information disclosure. To ensure a secure posture, both network and host access levels must be controlled.

Security Approaches

An organization can take multiple approaches to address the protection of its networks: either ignore security issues, provide host security, provide network-level security, or provide a combination of the latter two. The middle two, host security and network-level security, have prevention as well as detection and response components. Rather than view these two approaches as independent solutions, a mature organization uses both in a complementary fashion.

If an organization decides to ignore security, it has chosen to utilize the minimal amount of security that is provided with its workstations, servers, and devices. No additional security measures will be implemented. Each "out-of-the-box" system has certain security settings that can be configured, and they should be. Actually protecting an entire network, however, requires work in addition to the few protection mechanisms that come with systems by default.

Host Security

Host security takes a granular view of security by focusing on protecting each computer and device individually instead of addressing protection of the network as a whole. When host security is used, each computer is relied upon to protect itself. If an organization decides to implement only host security and does not include network security, there is a high

probability of introducing or overlooking vulnerabilities. Most environments are filled with different operating systems (Windows, UNIX, Linux, macOS), different versions of those operating systems, and different types of installed applications. Each operating system has security configurations that differ from those of other systems, and different versions of the same operating system may in fact have configuration variations between them.

Host security is important and should always be addressed. Security, however, should not stop there, as host security is a complementary process to be combined with network security. If individual host computers have vulnerabilities embodied within them, then network security can provide another layer of protection that will, hopefully, stop any intruders who have gotten that far into the environment.

A longtime discussion has centered on whether host- or network-based security is more important. Most security experts now generally agree that a combination of both is needed to adequately address the wide range of possible security threats. Certain attacks are more easily spotted, and some attacks are more easily prevented using tools designed for one or the other of these approaches.

Network Security

In some smaller environments, host security by itself may be an option, but as systems become connected into networks, security should include the actual network itself. In **network security**, an emphasis is placed on controlling access to internal computers from external entities. This control can be through devices such as routers, firewalls, authentication hardware and software, encryption, and intrusion detection systems (IDSs).

Network environments tend to be unique entities because usually no two networks have exactly the same number of computers, the same applications installed, the same number of users, the exact same configurations, or the same available servers. They will not perform the same functions or have the same overall architecture. Since networks have so many variations, there are many different ways in which they can be protected and configured. This chapter covers some foundational approaches to network and host security. Each approach may be implemented in a myriad of ways, but both network and host security need to be addressed for an effective total security program.

Security Principles

In the mid-1970s, two computer scientists from MIT, Jerome Saltzer and Michael Schroeder, published a paper on design principles for a secure computer system. The Saltzer and Schroeder paper, titled "The Protection of Information in Computer Systems," has been hailed as a seminal work in computer security, and the eight design principles in it are as relevant today as they were in 1970s. These principles are useful in secure system design and operation. These eight principles form the foundation for the field. Since they were published, an additional five have been adopted, and this set has stood for decades.

Least Privilege

One of the most fundamental principles in security is **least privilege**. This concept is applicable to many physical environments as well as network and host security. Least privilege means that a subject (which may be a user, application, or process) should have only the necessary rights and privileges to perform its task, with no additional permissions. Limiting an

object's privileges limits the amount of harm that can be caused, thus limiting an organization's exposure to damage. Users may have access to the files on their workstations and a select set of files on a file server, but no access to critical data that is held within the database. This rule helps an organization protect its most sensitive resources and helps ensure that whoever is interacting with these resources has a valid reason to do so.

Least privilege means that a subject should have only the necessary rights and privileges to perform its task, with no additional permissions.

Try This!

Examples of the Least Privilege Principle

The security concept of least privilege is not unique to computer security. It has been practiced by organizations such as financial institutions and governments for centuries. Basically, it simply means that individuals are given only the absolute minimum of privileges that are required to accomplish their assigned job. Examine the security policies that your organization has in place and see if you can identify examples of where the principle of least privilege has been used.

The concept of least privilege applies to more network security issues than just providing users with specific rights and permissions. When trust relationships are created, they should not be implemented in such a way that everyone trusts each other simply because it is easier. One domain should trust another for very specific reasons, and the implementers should have a full understanding of what the trust relationship allows between two domains. If one domain trusts another, do all of the users automatically become trusted, and can they thus easily access any and all resources on the other domain? Is this a good idea? Is there a more secure way of providing the same functionality? If a trusted relationship is implemented such that users in one group can access a plotter or printer that is available in only one domain, it might make sense to simply purchase another device so that other, more valuable or sensitive resources are not accessible by the entire group.

Another issue that falls under the least privilege concept is the security context in which an application runs. All applications, scripts, and batch files run in the security context of a specific user on an operating system. They execute with specific permissions as if they were a user. The application may be Microsoft Word and run in the space of a regular user, or it may be a diagnostic program that needs access to more sensitive system files and so must run under an administrative user account, or it may be a program that performs backups and so should operate within the security context of a backup operator. The crux of this issue is that a program should execute only in the security context that is needed for that program to perform its duties successfully. In many environments, people do not really understand how to make programs run under different security contexts, or it may just seem easier to have all programs run under the administrator account. If attackers can compromise a program or service running under the administrator account, they have effectively elevated their access level and have much more control over the system and many more ways to cause damage.

Try This!

Control of Resources

Being able to apply the appropriate security control to file and print resources is an important aspect of the least privilege security principle. How this is implemented varies depending on the operating system the computer runs. Check how the operating system you use provides for the ability to control file and print resources.

Separation of Privilege

Protection mechanisms can be employed to grant access based on a variety of factors. One of the key principles is to base decisions on more than a single piece of information. The principle of **separation of privilege** states that the protection mechanism should be constructed so that it uses more than one piece of information to make access decisions. Applying this principle to the people side of the security function results in the concept of **separation of duties**.

The principle of separation of privilege is applicable to physical environments as well as network and host security. When applied to people's actions, separation of duties specifies that for any given task, more than one individual needs to be involved. The task is broken into different duties, each of which is accomplished by a separate individual. By implementing a task in this manner, no single individual can abuse the system for their own gain. This principle has been implemented in the business world, especially financial institutions, for many years. A simple example is a system in which one individual is required to place an order and a separate person is needed to authorize the purchase.

Separation of duties specifies that for any given task, more than one individual needs to be involved.

While separation of duties provides a certain level of checks and balances, it is not without its own drawbacks. Chief among these is the cost required to accomplish the task. This cost is manifested in both time and money. More than one individual is required when a single person could accomplish the task, thus potentially increasing the cost of the task. In addition, with more than one individual involved, a certain delay can be expected because the task must proceed through its various steps.

Fail-Safe Defaults

Today, the Internet is no longer the friendly playground of researchers that it once was. This has resulted in different approaches that might at first seem less than friendly but that are required for security purposes. The concept of **fail-safe defaults** is that when something fails, it should do so to a safe state. One approach is that a protection mechanism should deny access by default and should grant access only when explicit permission exists. This is sometimes called **default deny**, and the common operational term for this approach is **implicit deny**.

Frequently in the network world, administrators make many decisions concerning network access. Often a series of rules will be used to determine whether or not to allow access (which is the purpose of a network firewall). If a particular situation is not covered by any of the other rules, the implicit deny approach states that access should not be granted. In other words,

if no rule would allow access, then access should not be granted. Implicit deny applies to situations involving both authorization and access.

The alternative to implicit deny is to allow access unless a specific rule forbids it. Another example of these two approaches is in programs that monitor and block access to certain websites. One approach is to provide a list of specific sites that a user is *not* allowed to access. Access to any site not on the list would be implicitly allowed. The opposite approach (the implicit deny approach) would block all access to sites that are not specifically identified as authorized. As you can imagine, depending on the specific application, one or the other approach will be more appropriate. Which approach you choose depends on the security objectives and policies of your organization.

Implicit deny is another fundamental principle of security, and students need to be sure they understand this principle. Similar to least privilege, this principle states that if you haven't specifically been allowed access, it should be denied.

Economy of Mechanism

The terms *security* and *complexity* are often at odds with each other, because the more complex something is, the harder it is to understand, and you cannot truly secure something if you do not understand it. Another reason complexity is a problem within security is that it usually allows too many opportunities for something to go wrong. If an application has 4000 lines of code, there are a lot fewer places for buffer overflows, for example, than in an application of two million lines of code. The principle of **economy of mechanism** is described as always using simple solutions when available.

Keep it simple. Another method of looking at the principle of economy of mechanism is that the protection mechanism should be small and simple.

An example of this principle concerns the number of services you allow your system to run. Default installations of computer operating systems often leave many services running. The keep-it-simple principle tells us to eliminate or disable those services we don't need. This is also a good idea from a security standpoint because it results in fewer applications that can be exploited and fewer services that the administrator is responsible for securing. The general rule of thumb is to eliminate or disable all nonessential services and protocols. This, of course, leads to the question, how do you determine whether or not a service or protocol is essential? Ideally, you should know what your computer system or network is being used for, and thus you should be able to identify and activate only those elements that are essential. For a variety of reasons, this is not as easy as it sounds. Alternatively, a stringent security approach that one can take is to assume that no service is necessary (which is obviously absurd) and activate services and ports only as they are requested. Whatever approach is taken, there is a never-ending struggle to try to strike a balance between providing functionality and maintaining security.

Complete Mediation

One of the fundamental tenets of a protection system is to check all access requests for permission. Each and every time a subject requests access to an object, the permission must be checked; otherwise, an attacker might gain unauthorized access to an object. **Complete mediation** refers to the concept that each and every request should be verified. When permissions are verified the first time, and the result is cached for subsequent use, performance may be increased, but this also opens the door to permission errors. Should a permission change subsequent to the first use, this change would not be applied to the operations after the initial check.

Complete mediation also refers to ensuring that all operations go through the protection mechanism. When security controls are added after the fact, it is important to make certain that all process flows are covered by the controls, including exceptions and out-of-band requests. If an automated process is checked in one manner, but a manual paper backup process has a separate path, it is important to ensure all checks are still in place. When a system undergoes disaster recovery or business continuity processes, or backup and/or restore processes, these too require complete mediation.

Open Design

The principle of **open design** holds that the protection of an object should not rely upon secrecy of the protection mechanism itself. This principle has been long proven in cryptographic circles, where hiding the algorithm ultimately fails and the true protection relies upon the secrecy and complexity of the keys. The principle does not exclude the idea of using secrecy, but merely states that, on the face of it, secrecy of mechanism is not sufficient for protection.

Another concept in security that should be discussed in this context is the idea of **security through obscurity**. In this case, security is considered effective if the environment and protection mechanisms are confusing or thought to be not generally known. Security through obscurity uses the approach of protecting something by hiding it. Non-computer examples of this concept include hiding your briefcase or purse if you leave it in the car so that it is not in plain view, hiding a house key under a doormat or in a planter, and pushing your favorite ice cream to the back of the freezer so that everyone else thinks it is all gone. The idea is that if something is out of sight, it is out of mind. This approach, however, does not provide actual protection of the object. Someone can still steal the purse by breaking into the car, lift the doormat and find the key, or dig through the items in the freezer to find your favorite ice cream. Security through obscurity may make someone work a little harder to accomplish a task, but it does not prevent anyone from eventually succeeding.

Similar approaches are seen in computer and network security when attempting to hide certain objects. A network administrator may, for instance, move a service from its default port to a different port so that others will not know how to access it as easily, or a firewall may be configured to hide specific information about the internal network in the hope that potential attackers will not obtain the information for use in an attack on the network.

In most security circles, security through obscurity is considered a poor approach, especially if it is the only approach to security. Security through obscurity simply attempts to hide an object; it doesn't implement a security control to protect it. An organization can use security through obscurity measures to try to hide critical assets, but other security measures should also be employed to provide a higher level of protection. For example, if an administrator moves a service from its default port to a more obscure port, an attacker can still actually find this service; thus, a firewall should be used to restrict access to the service. Most people know that even if you do shove your ice cream to the back of the freezer, someone may eventually find it.

Tech Tip

Security Through Obscurity

The principle of open design and the practice of security by obscurity may seem at odds with each other, but in reality they are not. The principle of open design states that secrecy itself cannot be relied upon as a means of protection. The practice of security through obscurity is a proven method of increasing the work factor that an adversary must expend to successfully attack a system. By itself, obscurity is not good protection, but it can complement other controls when both are properly employed.

It often amazes security professionals how frequently individuals rely on security through obscurity as their main line of defense. Relying on some piece of information remaining secret is generally not a good idea. This is especially true in this age of reverse engineering, where individuals analyze the binaries for programs to discover embedded passwords or cryptographic keys. The biggest problem with relying on security through obscurity is that if it fails and the secret becomes known, there often is no easy way to modify the secret to secure it again.

Tech Tip

Sandboxing

Sandboxing separates applications from critical operating systems services and other programs. It adds an extra layer of security to protect critical functions from malware or infected apps.

Least Common Mechanism

The principle of **least common mechanism** states that mechanisms used to access resources should be dedicated and not shared. Sharing of mechanisms allows a potential crossover between channels, resulting in a protection failure mode. For example, if there is a module that enables employees to check their payroll information, a separate module should be employed to change the information, lest a user gain access to change versus read access. Although sharing and reuse are good in one sense, they can represent a security risk in another.

Common examples of the least common mechanism and its isolation principle abound in ordinary systems. *Sandboxing* is a means of separating the operation of an application from the rest of the operating system. Virtual machines (VMs) perform the same task between operating systems on a single piece of hardware. Instantiating shared libraries, in which separate instantiation of local classes enables separate but equal coding, is yet another. The key is to provide a means of isolation between processes so information cannot flow between separate users unless specifically designed to do so.

Psychological Acceptability

Psychological acceptability refers to the users' acceptance of security measures. Another name for psychological acceptability is *least astonishment*, referring to the role that security measures should play with respect to *usability*. Users play a key role in the operation of a system, and if security measures are perceived to be an impediment to the work a user is responsible for, then a natural consequence may be that the user bypasses the control. Although a user may understand that this could result in a security problem, the perception that it does result in their performance failure will present pressure to bypass it.

Psychological acceptability is often overlooked by security professionals focused on technical issues and how they see the threat. They are focused on the threat, which is their professional responsibility, so the focus on security is natural and aligns with their professional responsibilities. This alignment between security and professional work responsibilities does not always translate to other positions in an organization. Security professionals, particularly those designing the security systems, should not only be aware of this concept but should pay particular attention to how security controls will be viewed by workers in the context of their work responsibility, not with respect to security for its own sake.

Defense in Depth

Defense in depth is a principle that is characterized by the use of multiple, different defense mechanisms with a goal of improving the defensive response to an attack. Another term for defense in depth is **layered security**. Single points of failure represent just that—an opportunity to fail. By using multiple defenses that are different, with differing points of failure, a system becomes stronger. While one defense mechanism may not be 100 percent effective, the application of a second defense mechanism to the items that succeed in bypassing the first mechanism provides a stronger response. A couple of different mechanisms can be employed in a defense-in-depth strategy: layered

security and diversity of defense. Together these provide a defense-in-depth strategy that is stronger than any single layer of defense.

A bank does not protect the money that it stores only by using a vault. It has one or more security guards as a first defense to watch for suspicious activities and to secure the facility when the bank is closed. It may have monitoring systems that watch various activities that take place in the bank, whether involving customers or employees. The vault is usually located in the center of the facility, and thus there are layers of rooms or walls before arriving at the vault. There is **access control**, which ensures that the people entering the vault have to be given authorization beforehand. And the systems, including manual switches, are connected directly to the police station in case a determined bank robber successfully penetrates any one of these layers of protection.

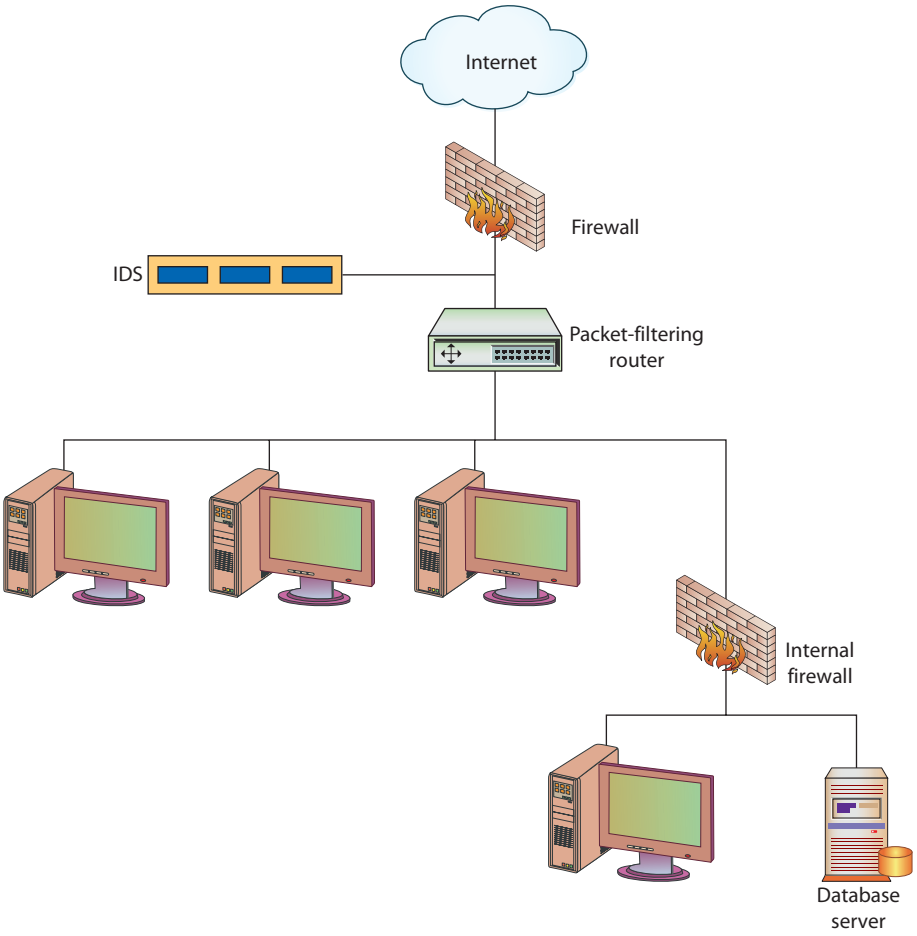
Networks should utilize the same type of layered security architecture. There is no "100 percent secure" system, and there is nothing that is foolproof, so a single specific protection mechanism should never be solely relied upon. It is important that every environment have multiple layers of security. These layers may employ a variety of methods, such as routers, firewalls, network segments, IDSs, encryption, authentication software, physical security, and traffic control. The layers need to work together in a coordinated manner so that one does not impede another's functionality and introduce a security hole.

As an example, consider the steps an intruder might have to take to access critical data held within a company's back-end database. The intruder first has to penetrate the firewall and use packets and methods that will not be identified and detected by the IDS (more information on these devices can be found in Chapter 13). The attacker next has to circumvent an internal router performing packet filtering, and then possibly penetrate another firewall used to separate one internal network from another (see Figure 2.3). From there, the intruder must break the access controls that are on the database, which means having to do a dictionary or brute force attack to be able to authenticate to the database software. Once the intruder has gotten this far, the data still needs to be located within the database. This may in turn be complicated by the use of access control lists outlining who can actually view or modify the data. That is a lot of work.

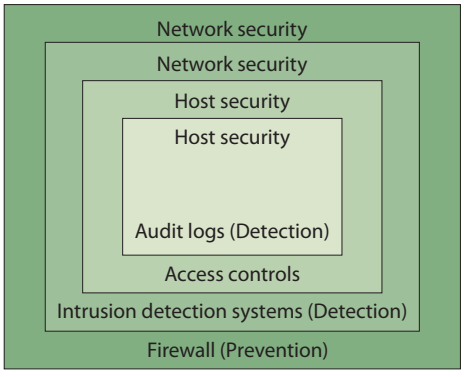
This example illustrates the different layers of security many environments employ. It is important to implement several different layers because if intruders succeed at one layer, you want to be able to stop them at the next. The redundancy of different protection layers ensures that there is no one single point of failure pertaining to security. If a network used only a firewall to protect its assets, an attacker able to penetrate this device successfully would find the rest of the network open and vulnerable.

An example of how different security methods can work against each other is when firewalls encounter encrypted network traffic. An organization may utilize encryption so that an outside customer communicating with a specific web server is assured that sensitive data being exchanged is protected. If this encrypted data is encapsulated within Secure Sockets Layer (SSL) or Transport Layer Security (TLS) packets and then sent through a firewall, the firewall may not be able to read the payload information in the individual packets.

Defense in depth can extend beyond simple technical measures. Using different vendors (vendor diversity) provides safeguards against vendor supply issues and vendor-specific technical issues.



• **Figure 2.3** Layered security



• **Figure 2.4** Various layers of security

The layers usually are depicted starting at the top, with more general types of protection, and progressing downward through each layer, with increasing granularity at each layer as you get closer to the actual resource, as you can see in Figure 2.4. This is because the top-layer protection mechanism is responsible for looking at an enormous amount of traffic, and it would be overwhelming and cause too much of a performance degradation if each aspect of the packet were inspected. Instead, each layer usually digs deeper into the packet and looks for specific items. Layers that are closer to the resource have to deal with only a fraction of the traffic that the top-layer security mechanism does, and thus looking deeper and at more granular aspects of the traffic will not cause as much of a performance hit.

Diversity of Defense

Diversity of defense is a concept that complements the idea of various layers of security. It involves making different layers of security dissimilar so that even if attackers know how to get through a system that comprises one layer, they may not know how to get through a different type of layer that employs a different system for security.

If an environment has two firewalls that form a demilitarized zone (DMZ), for example, one firewall may be placed at the perimeter of the Internet and the DMZ. This firewall analyzes the traffic that is entering through that specific access point and enforces certain types of restrictions. The other firewall may then be placed between the DMZ and the internal network. When applying the diversity-of-defense concept, you should set up these two firewalls to filter for different types of traffic and provide different types of restrictions. The first firewall, for example, may make sure that no FTP, SNMP, or Telnet traffic enters the network but allow SMTP, SSH, HTTP, and SSL traffic through. The second firewall may not allow SSL or SSH through and may interrogate SMTP and HTTP traffic to make sure that certain types of attacks are not part of that traffic.

Encapsulation

The principle of **encapsulation** is used all of the time in protocols. When a higher-level protocol is used to carry a lower protocol, the lower protocol is encapsulated in the data portion of the higher protocol. Think of it like an envelope within an envelope. This simple concept allows separate protocols to work with each other, without interference, and without needing to understand the material being encapsulated. The Open System Interconnection (OSI) and Internet models separating functions into different layers—from the physical layer to the data layer to the network layer, and so on—is an example of encapsulation.

Isolation

Isolation is the concept of separating items so that they cannot interfere with each other. Isolation is common in many functions, both in hardware and software, with the express purpose of preventing interference between the separate processes. Examples of isolation include confinement of a program in a sandbox, or a virtual machine, system call interposition, and software fault isolation.

Trust Relationships

Trust is defined as having an understanding of how a party will react to a given stimulus. In other words, if X happens, what will the party do in response? In this regard, you can trust a scorpion to sting you, even if you rescue him from flood waters. Why? Because that is what scorpions do when handled. Trust is also a key principle in computer security. Will you share resources with another user? The answer depends upon a trust relationship. If you establish a trust relationship between systems, you are granting access to another user or set of resources to perform specific tasks associated with your resources.

Changes in trust occur at **trust boundaries**—logical boundaries that surround specific levels of trust in a system. When outside input is entered into a computer program, it is crossing a trust boundary, and a decision has to be made as to whether or not the input should be trusted. Another name for the boundary around a system where external inputs can interact with a system is referred to as the **attack surface**. A key element in limiting hostile inputs is attack surface minimization, or the limiting of trusting outside information.

Tech Tip

Security Design Principles

The eight design principles from Saltzer and Schroeder are listed and paraphrased here:

- ***Least privilege*** *Use minimum privileges necessary to perform a task.*
- ***Separation of privilege*** *Access should be based on more than one item.*
- ***Fail-safe defaults*** *Deny by default (implicit deny) and only grant access with explicit permission.*
- ***Economy of mechanism*** *Mechanisms should be small and simple.*
- ***Complete mediation*** *Protection mechanisms should cover every access to every object.*
- ***Open design*** *Protection mechanisms should not depend on the secrecy of the mechanism itself.*
- ***Least common mechanism*** *Protection mechanisms should be shared to the least degree possible among users.*
- ***Psychological acceptability*** *Protection mechanisms should not impact users, but if they do, the impact should be minimal.*

Reference: J.H. Saltzer and M.D. Schroeder, "The Protection of Information in Computer Systems," Proc. IEEE, vol. 63, no. 9, 1975, pp. 1278–1308.

Many security failures can be traced to a problem with trust. Social engineering, where someone pretends to be someone they are not, is a trust violation—one that preys on customer service's desire to be helpful. Losing control over internal users, allowing them to have access to more than they need, can create trust issues. Imagine if every user had a building master key—if something ends up missing, then everyone with a key becomes suspect. And to further complicate the issue, one needs to remember that trust can be transitive and shared with other parties. You may trust your assistant with a key, but do you trust everyone they might loan it to? When one applies this line of thinking to the myriad of trust relationships within a data system—developers, customers, third parties, and vendors—these relationships become a challenge. For example, a network trust failure allowed an attacker to get to Target's point-of-sale system via a third-party heating, ventilation, and air conditioning (HVAC) vendor, resulting in a large data breach.

Because of the nature of trust and its high-risk opportunity, the sage advice is to develop and maintain a culture of *reluctance to trust*. For systems to work between parties, trust must be shared, but the sharing should be limited and controlled to only that which is needed for business purposes. Excessive trust only increases risk, without any business benefit.

Formal Security Models

An important issue when designing the software that will operate and control secure computer systems and networks is the security model that the system or network will be based on. The security model will implement the chosen security policy and enforce those characteristics deemed most important by the system designers. For example, if confidentiality is considered paramount, the model should make certain no data is disclosed to unauthorized individuals. A model enforcing confidentiality may allow unauthorized individuals to modify or delete data, as this would not violate the tenets of the model because the true values for the data would still remain confidential. Of course, this model might not be appropriate for all environments. In some instances, the unauthorized modification of data may be considered a more serious issue than its unauthorized disclosure. In such cases, the model would be responsible for enforcing the integrity of the data instead of its confidentiality. Choosing the model to base the design on is critical if you want to ensure that the resultant system accurately enforces the security policy desired. This, however, is only the starting point, and it does not imply that you have to make a choice between confidentiality and data integrity, as both are important.

Confidentiality Models

Data confidentiality has generally been the chief concern of the military. For instance, the U.S. military encouraged the development of the Bell-LaPadula security model to address data confidentiality in computer operating systems. This model is especially useful in designing multilevel security systems that implement the military's hierarchical security scheme, which

includes levels of classification such as Unclassified, Confidential, Secret, and Top Secret. Similar classification schemes can be used in industry, where classifications might include Publicly Releasable, Proprietary, and Company Confidential.

A second confidentiality model, the Brewer-Nash security model, is one defined by controlling read and write access based on conflict of interest rules. This model is also known as the Chinese Wall model, after the concept of separating groups through the use of an impenetrable wall.

Bell-LaPadula Model

The **Bell-LaPadula security model** employs both mandatory and discretionary access control mechanisms when implementing its two basic security principles. The first of these principles is called the **Simple Security Rule**, which states that no subject (such as a user or a program) can read information from an object (such as a file) with a security classification higher than that possessed by the subject itself. This means that the system must prevent a user with only a Secret clearance, for example, from reading a document labeled Top Secret. This rule is often referred to as the "no-read-up" rule.

The Simple Security Rule is just that: the most basic of security rules. It essentially states that in order for you to see something, you have to be authorized to see it.

The second security principle enforced by the Bell-LaPadula security model is known as the ***-property** (pronounced *star property*). This principle states that a subject can write to an object only if the target's security classification is greater than or equal to the object's security classification. This means that a user with a Secret clearance can write to a file with a Secret or Top Secret classification but cannot write to a file with only an Unclassified classification. This at first may appear to be a bit confusing, since this principle allows users to write to files that they are not allowed to view, thus enabling them to actually destroy files that they don't have the classification to see. This is true, but keep in mind that the Bell-LaPadula model is designed to enforce confidentiality, not integrity. Writing to a file that you don't have the clearance to view is not considered a confidentiality issue; it is an integrity issue.

Whereas the *-property allows a user to write to a file of equal or greater security classification, it doesn't allow a user to write to a file with a lower security classification. This, too, may be confusing at first—after all, shouldn't a user with a Secret clearance, who can view a file marked Unclassified, be allowed to write to that file? The answer to this, from a security perspective, is "no." The reason again relates to wanting to avoid either accidental or deliberate security disclosures. The system is designed to make it impossible (hopefully) for data to be disclosed to those without the appropriate level to view it. As shown in Figure 2.5, if it were possible for a user with a Top Secret clearance to either deliberately or accidentally write Top Secret information and place it in a file marked Confidential, a user with only a Confidential security clearance could then access this file and view the Top Secret information. Thus, data would have been disclosed to an individual not authorized to view it. This is what the system should protect against and is the reason for what is known as the "no-write-down" rule.

Not all environments are more concerned with confidentiality than integrity. In a financial institution, for example, viewing somebody's bank balance is an issue, but a greater issue would be

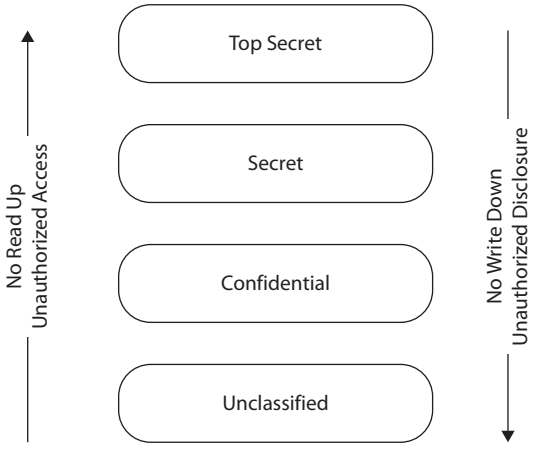
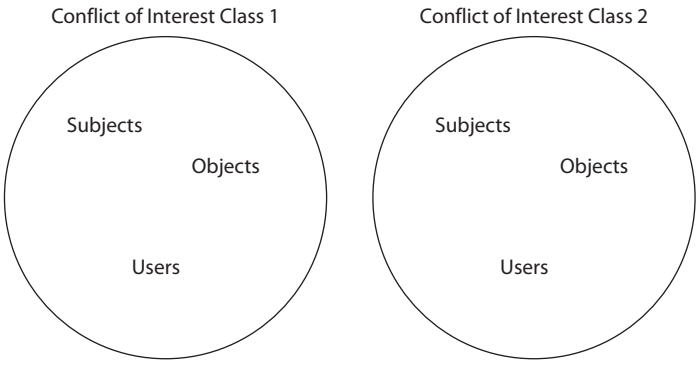


Figure 2.5 Bell-LaPadula security model

the ability to actually modify that balance. In environments where integrity is more important, a different model than the Bell-LaPadula security model is needed.

Brewer-Nash Security Model

One of the tenets associated with access is *need to know*. Separate groups within an organization may have differing needs with respect to access to information. A security model that takes into account user conflict-of-interest aspects is the **Brewer-Nash security model**. In this model, information flows are modeled to prevent information from flowing between subjects and objects when a conflict of interest would occur. As previously noted, this model is also known as the Chinese Wall model, after the Great Wall of China, a structure designed to separate groups of people. As shown in Figure 2.6, separate groups are defined and access controls are designed to enforce the separation of the groups.



• **Figure 2.6** Brewer-Nash security model

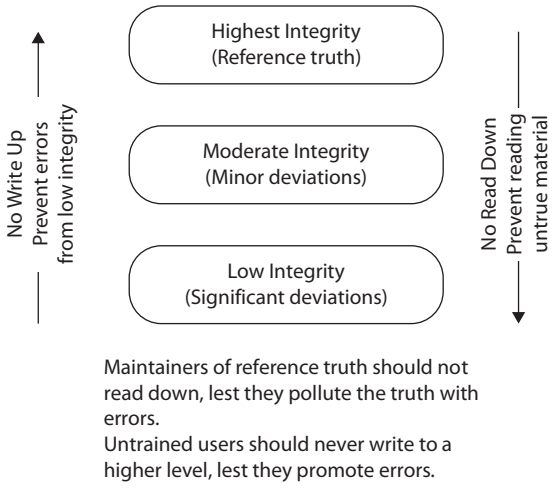
Integrity Models

The Bell-LaPadula model was developed in the early 1970s but was found to be insufficient for all environments. As an alternative, Kenneth Biba studied the integrity issue and developed what is called the Biba security model in the late 1970s. Additional work was performed in the 1980s that led to the Clark-Wilson security model, which also places its emphasis on integrity rather than confidentiality.

The Biba Security Model

In the **Biba security model**, shown in Figure 2.7, instead of security classifications, *integrity levels* are used. A principle of integrity levels is that data with a higher integrity level is believed to be more accurate or reliable than data with a lower integrity level. Integrity levels indicate the amount of "trust" that can be placed in information at the different levels. Integrity levels differ from security levels in another way—they limit the modification of information as opposed to the flow of information.

An initial attempt at implementing an integrity-based model was captured in what is referred to as the **Low-Water-Mark policy**. This policy in many ways is the opposite of the *-property in that it prevents subjects from writing to objects of a higher integrity level. The policy also contains a second rule that states the integrity level of a subject will be lowered if it reads an object of a lower integrity level. The reason for this is that if the subject then uses data from that object, the highest the integrity level can be for a new object created from it is the same level of integrity as the original object. In other words, the level of trust you can place in data formed from data at a specific integrity level cannot be higher than the level of trust you have in the subject creating the new data object, and the level of trust you have in



• **Figure 2.7** Biba security model

the subject can only be as high as the level of trust you had in the original data. The final rule contained in the Low-Water-Mark policy states that a subject can execute a program only if the program's integrity level is equal to or less than the integrity level of the subject. This ensures that data modified by a program only has the level of trust (integrity level) that can be placed in the individual who executed the program.

While the Low-Water-Mark policy certainly prevents unauthorized modification of data, it has the unfortunate side effect of eventually lowering the integrity levels of all subjects to the lowest level on the system (unless the subject always views files with the same level of integrity). This is because of the second rule, which lowers the integrity level of the subject after accessing an object of a lower integrity level. There is no way specified in the policy to ever raise the subject's integrity level back to its original value. A second policy, known as the **Ring policy**, addresses this issue by allowing any subject to read any object without regard to the object's level of integrity and without lowering the subject's integrity level. This, unfortunately, can lead to a situation where data created by a subject after reading data of a lower integrity level could end up having a higher level of trust placed upon it than it should.

The Biba security model implements a hybrid of the Ring and Low-Water-Mark policies. Biba's model in many respects is the opposite of the Bell-LaPadula model in that what it enforces are "no-read-down" and "no-write-up" policies. It also implements a third rule that prevents subjects from executing programs of a higher level. The Biba security model thus addresses the problems mentioned with both the Ring and Low-Water-Mark policies.

The Clark-Wilson Security Model

The **Clark-Wilson security model** takes an entirely different approach than the Biba and Bell-LaPadula models, using transactions as the basis for its rules. It defines two levels of integrity only: constrained data items (CDIs) and unconstrained data items (UDIs). CDI data is subject to integrity controls, whereas UDI data is not. The model then defines two types of processes: integrity verification processes (IVPs), which ensure that CDI data meets integrity constraints (to ensure the system is in a valid state), and transformation processes (TPs), which change the state of data from one valid state to another. Data in this model cannot be modified directly by a user; it must be changed by trusted TPs, access to which can be restricted (thus restricting the ability of a user to perform certain activities).

It is useful to return to the prior example of the banking account balance to describe the need for integrity-based models. In the Clark-Wilson model, the account balance would be a CDI because its integrity is a critical function for the bank. A client's color preference for their checkbook is not a critical function and would be considered a UDI. Since the integrity of account balances is of extreme importance, changes to a person's balance must be accomplished through the use of a TP. Ensuring that the balance is correct would be the duty of an IVP. Only certain employees of the bank should have the ability to modify an individual's account, which can be controlled by limiting the number of individuals who have the authority to execute TPs that result in account modification. Certain very critical functions may actually be split into multiple TPs to enforce another important

principle: *separation of duties* (introduced earlier in the chapter). This limits the authority any one individual has so that multiple individuals will be required to execute certain critical functions.

Additional References

Saltzer, J.H., and M.D. Schroeder. 1975. "The Protection of Information in Computer Systems." *Proc. IEEE* 63(9): 1278–1308.

Schwartau, W. 1998. "Time-Based Security Explained: Provable Security Models and Formulas for the Practitioner and Vendor." *Computers & Security* 17(8): 693–714.

Chapter 2 Review

Chapter Summary

After reading this chapter and completing the exercises, you should understand the following regarding the basics of security, security terminology, and security models.

Define basic terms associated with computer and information security

- Information assurance and information security place the security focus on the information and not on the hardware or software used to process it.
- The original goal of computer and network security was to provide confidentiality, integrity, and availability—the "CIA" of security.
- Additional elements of security can include authentication, authorization, auditability, and nonrepudiation.
- The operational model of computer security tells us that protection is provided by prevention, detection, and response.

Identify the basic approaches to computer and information security

- Host security focuses on protecting each computer and device individually, whereas network security focuses on addressing protection of the network as a whole.
- For many organizations, a combination of host security and network security is needed to adequately address the wide range of possible security threats.

Identify the basic principles of computer and information security

- The principle of least privilege states that the minimum privileges necessary to perform a task should be used.
- The principle of separation of privilege states that critical items should require multiple parties.
- The principle of fail-safe default states that access decisions should be based on deny by default (implicit deny) with specific exceptions being made via granting access using explicit permissions.
- The principle of economy of mechanism states that protection mechanisms should be small and simple.
- The principle of complete mediation states that protection mechanisms should cover every access to every object and should never be bypassed.
- The principle of open design states that protection mechanisms should not depend on the secrecy of the mechanism itself.
- The principle of least common mechanism states that the protection mechanisms should be shared to the least degree possible among users.
- The principle of psychological acceptability states that protection mechanisms should not impact users, but if they do, the impact should be minimal.
- The principle of defense in depth, or layered security, is that multiple layers of differing, overlapping controls should be employed.
- Diversity of defense is a concept that complements the idea of various layers of security. It means to make the layers dissimilar so that if one layer is penetrated, the next layer can't also be penetrated using the same method.

Recognize some of the basic models used to implement security in operating systems

- Security models enforce the chosen security policy.
- There are two basic categories of models: those that ensure confidentiality and those that ensure integrity.
- Bell-LaPadula is a confidentiality security model whose development was prompted by the demands of the U.S. military and its security clearance scheme. This security model enforces "no-read-up" and "no-write-down" rules to avoid the deliberate or accidental disclosure of information to individuals not authorized to receive it.
- The Brewer-Nash security model (the Chinese Wall model) is a confidentiality model that separates users based on conflicts of interest.
- The Biba security model is an integrity-based model that, in many respects, implements the opposite of what the Bell-LaPadula model does—that is, "no-read-down" and "no-write-up" rules.
- The Clark-Wilson security model is an integrity-based model designed to limit the processes an individual may perform as well as require that critical data be modified only through specific transformation processes.

Key Terms

***-property** *(43)*
access control *(39)*
attack surface *(41)*
auditability *(28)*
authentication *(28)*
availability *(28)*
Bell-LaPadula security model *(43)*
Biba security model *(44)*
Brewer-Nash security model *(44)*
Clark-Wilson security model *(45)*
complete mediation *(36)*
confidentiality *(28)*
default deny *(35)*
defense in depth *(38)*
diversity of defense *(40)*
economy of mechanism *(36)*
encapsulation *(41)*
fail-safe defaults *(35)*
fortress model *(28)*
hacking *(27)*
host security *(32)*
implicit deny *(35)*
integrity *(28)*
isolation *(41)*
layered security *(38)*
least common mechanism *(38)*
least privilege *(33)*
Low-Water-Mark policy *(44)*
network security *(33)*
nonrepudiation *(28)*
open design *(37)*
operational model of computer security *(29)*
phreaking *(27)*
psychological acceptability *(38)*
Ring policy *(45)*
security through obscurity *(37)*
separation of duties *(35)*
separation of privilege *(35)*
Simple Security Rule *(43)*
trust *(41)*
trust boundary *(41)*

Key Terms Quiz

Use terms from the Key Terms list to complete the sentences that follow. Don't use the same term more than once. Not all terms will be used.

1. _____ is a term used to describe the condition where a user cannot deny that an event has occurred.
2. The _____ is an integrity-based security model that bases its security on control of the processes that are allowed to modify critical data, referred to as *constrained data items.*
3. The security principle used in the Bell-LaPadula security model that states that no subject can read from an object with a higher security classification is called the _____.
4. The principle that states that a subject has only the necessary rights and privileges to perform its task, with no additional permissions, is called _____.
5. _____ is the principle in security where protection mechanisms should be kept as simple and as small as possible.
6. _____ is the principle that protection mechanisms should minimize user-level impact.
7. _____ is the process used to ensure that separate processes do not interfere with each other.
8. The architecture in which multiple methods of security defense are applied to prevent realization of threat-based risks is called _____.
9. _____ is the process of using multiple people to perform critical or sensitive tasks.
10. Implicit deny is an operationalization of the principle of _____.

Multiple-Choice Quiz

1. Which of the following is not a principle of security?
   - **A.** Principle of least privilege
   - **B.** Principle of economy of mechanism
   - **C.** Principle of efficient access
   - **D.** Principle of open access
2. The CIA of security includes which three security goals?
   - **A.** Confidentiality, integrity, authentication
   - **B.** Confidentiality, integrity, availability
   - **C.** Certificates, integrity, availability
   - **D.** Confidentiality, inspection, authentication
3. The security principle used in the Bell-LaPadula security model that states that no subject can read from an object with a higher security classification is known as what?
   - **A.** Simple Security Rule
   - **B.** Ring policy
   - **C.** Mandatory access control
   - **D.** *-property
4. Which of the following concepts requires users and system processes to use the minimal amount of permission necessary to function?
   - **A.** Layer defense
   - **B.** Diversified defense
   - **C.** Simple Security Rule
   - **D.** Least privilege

5. Which security model separates users based on conflict-of-interest issues?
   - **A.** Bell-LaPadula
   - **B.** Brewer-Nash
   - **C.** Biba
   - **D.** Clark-Wilson
6. The Bell-LaPadula security model is an example of a security model that is based on what?
   - **A.** The integrity of the data
   - **B.** The availability of the data
   - **C.** The confidentiality of the data
   - **D.** The authenticity of the data
7. What is the term used to describe the requirement that different portions of a critical process be performed by different people?
   - **A.** Least privilege
   - **B.** Defense in depth
   - **C.** Separation of duties
   - **D.** Job rotation
8. Hiding information to prevent disclosure is an example of what?
   - **A.** Security through obscurity
   - **B.** Certificate-based security
   - **C.** Discretionary data security
   - **D.** Defense in depth
9. What is the problem with the Low-Water-Mark policy?
   - **A.** It's aimed at ensuring confidentiality and not integrity.
   - **B.** It could ultimately result in all subjects having the integrity level of the least-trusted object on the system.
   - **C.** It could result in the unauthorized modification of data.
   - **D.** It does not adequately prevent users from viewing files they are not entitled to view.
10. The concept of blocking an action unless it is specifically authorized is known as what?
    - **A.** Implicit deny
    - **B.** Least privilege
    - **C.** Simple Security Rule
    - **D.** Hierarchical defense model

Essay Quiz

1. Your company has information that must be protected from unauthorized access, and it must also prevent certain business units in certain countries from accessing this information. Which security models apply and how?
2. The new CEO for your company just retired from the military and wants to use some of the same computer systems and security software she used while with the military. Explain to her the reasons that confidentiality-based security models are not adequate for all environments. Provide at least two examples of environments where a confidentiality-based security model is not sufficient.
3. Describe why the concept of "security through obscurity" is generally considered a bad principle to rely on. Provide some real-world examples of where you have seen this principle used.
4. Write a brief essay describing the principle of least privilege and how it can be employed to enhance security. Provide at least two examples of environments in which it can be used for security purposes.

Lab Project

• Lab Project 2.1

In an environment familiar to you (your school or where you work, for example), determine whether the principle of diversity of defense has been employed and list the different layers of security that are employed. Discuss whether you think they are sufficient and whether the principle of defense in depth has also been used.